**Thử nghiệm LBaas v2**

Contents

[I. Yêu cầu về LB đối với các giải pháp của Trung tâm 2](#_Toc508990513)

[II. LBaaS của openstack 2](#_Toc508990514)

[1. Tổng qua LBaas v1 vs LBaas v2 2](#_Toc508990515)

[2. Chi tiết về Octavia 4](#_Toc508990516)

[3. Đánh giá khả năng đáp ứng của Octavia với các giải pháp của Trung tâm: 7](#_Toc508990517)

[III. Hướng dẫn cài đặt và sử dụng Octavia trên cloud v1 8](#_Toc508990518)

[1. Hướng dẫn cài đặt: 8](#_Toc508990519)

[2. Hướng dẫn sử dụng: 12](#_Toc508990520)

# Yêu cầu về LB đối với các giải pháp của Trung tâm

Yêu cầu về LB đối với giải pháp antiddos waf layer 7:

* Giải pháp đang sử dụng hiện tại: haproxy
* Yêu cầu hoạt động ở chế độ transparent để backend phía sau là antiddos + waf sẽ nhìn thấy IP của client (Để thực hiện các tính năng giới hạn tốc độ connection/IP).
* Ngoài ra còn có 1 số cấu hình tối ưu thêm như sau:
  + Chạy ở chế độ active/active: Tuy nhiên khái niệm active/active của LB hiện tại cho antiddos + waf không phải khái niệm chuẩn của LB, bản chất là 2 IP active/standby kết hợp với nhau.
  + Chạy nhiều tiến trình haproxy, mỗi tiến trình bind vào 1 CPU.
  + Tối ưu các tham số của OS: max conntrack, max openfiles.
  + Cấu hình hash type: consistent để với mỗi 1 IP client sẽ đẩy vào 1 backend (antiddos + waf) cố định.

Yêu cầu về LB đối với giải pháp email security:

* Giải pháp LB hiện tại: dùng LVS (LB layer 4).
* Yêu cầu nhìn thấy được IP của client: Để đánh giá tốc độ gửi email => kiểm tra có spam hay không, kiểm tra các bản ghi SPF, DKIM, DMARC, ….

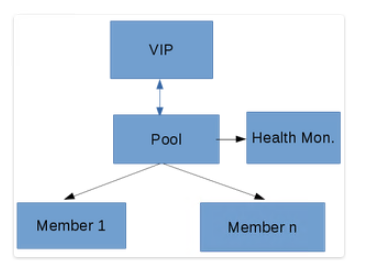
Như vậy yêu cầu quan trọng nhất về LB của các giải pháp của trung tâm đó là hoạt động ở chế độ transparent để các backend có thể nhìn thấy IP trực tiếp của client.

# LBaaS của openstack

# Tổng qua LBaas v1 vs LBaas v2

*LBaas v1:*

Mô hình hoạt động:



Luồng request: Request —> VIP —> Pool [Optional Health Monitor] —> Members [Backend instances]

Các tính năng đã có:

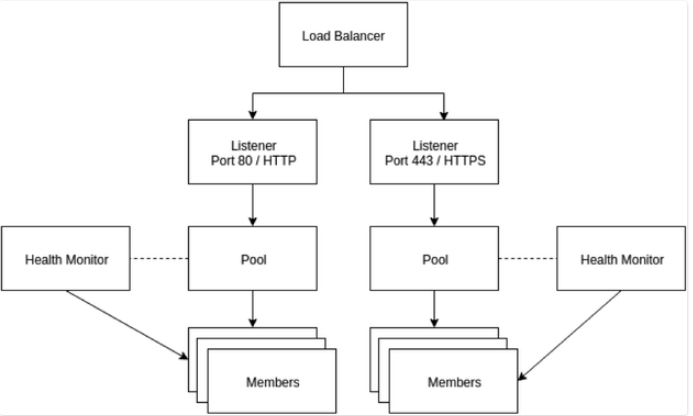
* L4 load balancing
* Session persistence including cookies based
* Cookie insertion
* Driver interface for 3rd parties
* Chỉ hỗ trợ hoạt động theo agent

Các tính năng còn thiếu (so về các tính năng phổ biến của LB)

* L7 Content switching [IMP feature]
* Multiple TCP ports per load balancer
* TLS Termination at load balancer to avoid the load on instances.
* Load balancer running inside instances.

*LBaasv2 (Từ bản kilo):*

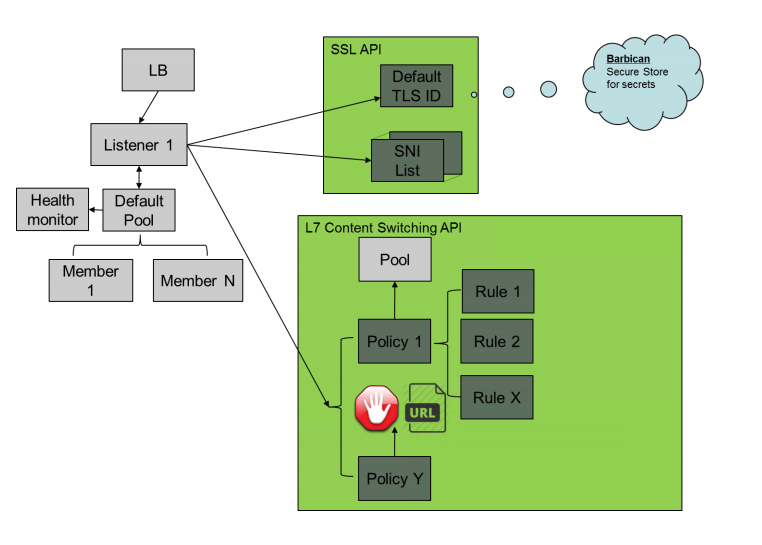
Mô hình hoạt động:



Luồng traffic: Request —> VIP —> Listeners –> Pool [Optional Health Monitor] —> Members [Backend instances]

Đã bổ sung thêm 1 số tính năng thường thấy trên LB:

* L7 content switching: Có thể tạo các rules theo domain, path, file type, … để forward traffic vào các pool backend khác nhau.
* Multiple TCP ports per load balancer: Đối với 1 LB thì có thể nhiều listener, ví dụ 1 listener cho port 80, 1 cho port 443. Các listener khác nhau có thể dùng chung 1 pool backend hoặc dùng các pool backend khác nhau. Còn trong Lbaas v1 thì không có khái niệm listener do bản chất 1 LB chỉ hỗ trợ 1 listener.
* TLS Termination at load balancer to avoid the load on instances: Có thể TLS termination tại LB để giảm tải cho backend phía sau. Để hoạt động được thì LBaas v2 kết hợp với Barbican để cung cấp tính năng TLS termination.

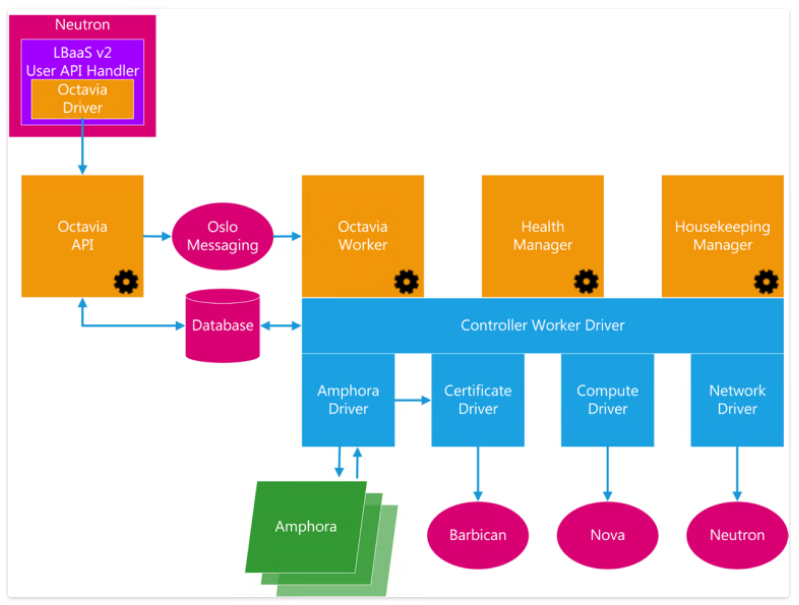


* Load balancer running inside instances: LBaasv2 có thể chạy ở chế độ agent trên network node hoặc ở chế độ VM instances (Octavia).
  + Agent: Chạy trên network node, agent sẽ quản lý cấu hình của dịch vụ haproxy.
  + Octavia: Chạy trên VM instances, bản chất cũng là haproxy tuy nhiên nếu dùng Octavia thì mô hình triển khai có thể linh hoạt hơn (ví dụ chạy chế độ single 1 VM hoặc active/standby 2 VM).

Do có nhiều ưu điểm hơn so với agent nên Octavia sẽ là hướng phát triển chính của LBaas v2 trong openstack.

# Chi tiết về Octavia

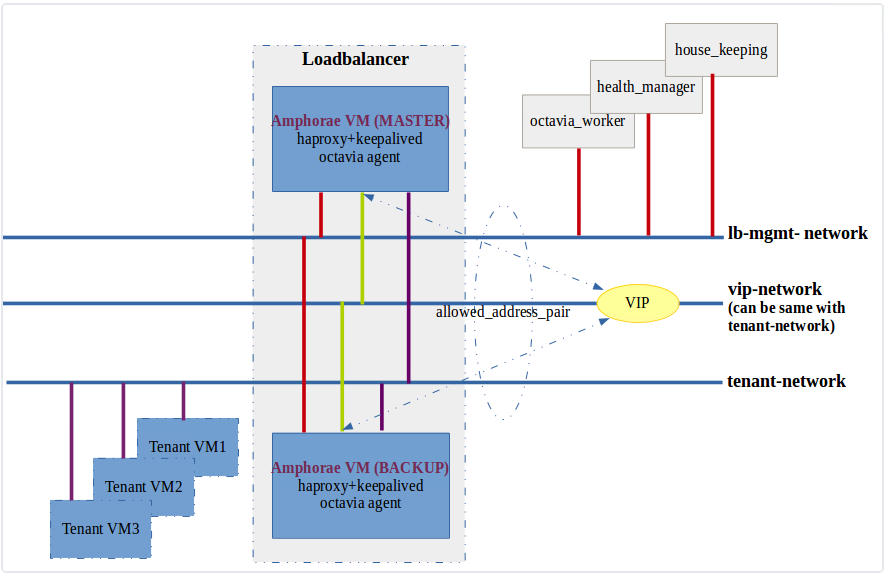
*Kiếm trúc của Octavia*



*Các thành phần chính của Octavia:*

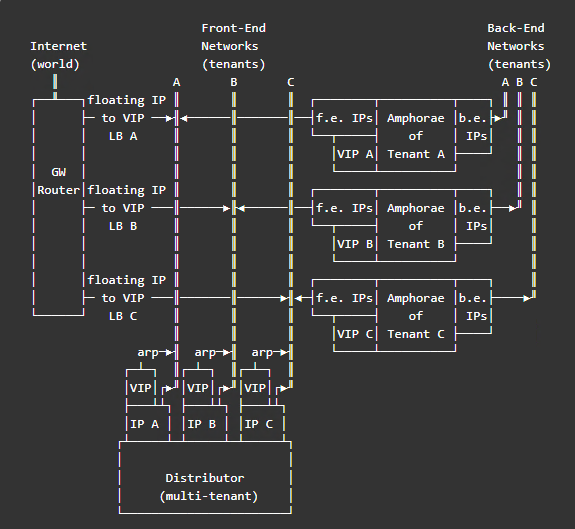
* Amphora: Là VM (Có thể chạy trên OS Ubuntu, fedora, ..) và cài haproxy, keepalived, octavia agent. Trong đó Haproxy: làm nhiệm vụ load balancer, keepalived phục vụ dự phòng cho LB trong mô hình active/standby, Octavia agent là để kết nối với thành phần của Octavia như Health manager, House Keeping, …
* Neutron-lbaas:
  + Trước pike (bao gồm cả ocata trên cloud v1): thì Octavia hoạt động kết hợp với neutron thông qua 1 Neutron plugin là neutron-lbaas. Khi đấy thì dữ liệu về LB tồn tại ở cả 2 database là neutron và Octavia => việc quản lý database sẽ phức tạp, không cần thiết.
  + Từ pike trở đi: Octavia đã hoạt động như 1 stand-alone service mà không cần dựa vào neutron.
* Octavia API: Giao tiếp trung gian giữa neutron và Octavia workers.
* Octavia Worker: Chịu trách nhiệm quản lý amphora như thêm, sửa xóa LB (VM), update cấu hình của LB.
* Octavia Health Manager: có 2 nhiệm vụ chính:
  + Lắng nghe dữ liệu gửi từ VM amphora để cập nhật trạng thái của LB, listener, pool, member => Cập nhật vào bảng amphora\_healthtable trong database. Ngoài ra cũng lắng nghe các thống kê (số connection đi qua LB) và cập nhật vào bảng listener\_statisticstable trong database.
  + Nếu 1 VM amphora bị lỗi thì Health Manager có trách nhiệm xóa VM cũ đi và thêm 1 VM amphora mới, vẫn dữ nguyên thông tin VIP.
* Octavia House Keeping: Định kỳ xóa thông tin amphora đã được xóa ra khỏi databases, update certificate cho amphora.
* Các Driver để liên lạc với các thành phần Neutron, Nova, Barbican, …

*Mô hình hoạt động của Lbaas v2 Octavia:*



* VM LB (Amphora) bao gồm các thành phần: haproxy +keepalived và Octavia agent.
* Octavia có thể hoạt động ở 2 topology:
  + Single: 1 VM chạy đơn. Khi đó sẽ không đảm bảo single point of failure.
  + Active/Standby: 2 VM chạy với 1 Virtual IP (Dùng keepalived) => Có thể đảm bảo dự phòng cho LB. Trong đó 1 VM amphora sẽ chạy chính, 1 VM amphora dự phòng.
* Lb-mgmt-network: Là mạng được dùng để kết nối Ocatavia Agent với các thành phần của Octavia lbaas như Octavia\_worker, health\_manager, house\_keeping
* Mạng tenant network: Là mạng nối với các backend.
* Mạng vip network: Có thể trùng hoặc không trùng với tenant network. Trong trường hợp VIP network không phải là mạng public thì cần float IP cho virtual IP để client truy cập từ ngoài vào.

Ngoài chế độ hoạt động theo mô hình single hoặc active/standby thì Ocata phiên bản mới đang phát triển mô hình active/active. Khi đấy các VM amphora cùng hoạt động đồng thời, và cần thêm 1 thành phần khác là Distributor để phân phối traffic cho các VM amphora. Distributor này hoạt động như 1 lớp Load balancer khác, và hoạt động tương tự như LVS ở chế độ direct return.



Theo như mô hình active/active này lại phát sinh nhược điểm là Distributor là single point of failures. Và mô hình active/active này cũng chỉ dùng 1 IP, khác với mô hình active/active đang chạy trên LB của antiddos waf, vì mô hình active/active của antiddos waf bản chất là kết hợp 2 active/standby của 2 IP.

# Đánh giá khả năng đáp ứng của Octavia với các giải pháp của Trung tâm:

Phiên bản hiện tại của Octavia thì chưa hỗ trợ tính năng transparent, và trong roadmap cũng không thấy có (Em đã từng đọc ở 1 tài liệu road map khác thì tính năng này có nhưng độ ưu tiên đang là thấp nhất).

<https://wiki.openstack.org/wiki/Octavia/Roadmap>

Do đó muốn sử dụng được LBaas v2 Octavia với các giải pháp của Trung tâm cần:

* Hỗ trợ transparent: Cần cấu hình iptables, ip route của OS và cấu hình trong haproxy
* Tối ưu hiệu năng:
  + Tối ưu hiệu năng cho haproxy: Chạy nhiều tiến trình haproxy, mỗi tiến trình map vào 1 CPU khác nhau.
  + Tối ưu hiệu năng cho OS: Tối ưu các tham số contrack max, open files.

Ngoài ra đối với mô hình active/active của LB hiện tại do đang chạy với mô hình không chuẩn, nên cũng không cần custom Octavia để chạy theo mô hình này, mà chỉ cần chạy theo mô hình active/standby là được.

Theo đánh giá thì phiên bản hiện tại đã có thể triển khai được với mô hình nhiều backend không cần yêu cầu transparent, và không cần hiệu năng quá cao.

# Hướng dẫn cài đặt và sử dụng Octavia trên cloud v1

# Hướng dẫn cài đặt:

*Bước 1: Tạo user Octavia trên openstack, tạo service, endpoint cho Octavia:*

Controller01

openstack user create --domain default --password-prompt octavia

openstack role add --project service --user octavia admin

openstack service create --name octavia --description "OpenStack Load Balancer" load-balancer

openstack endpoint create --region North\_VN octavia public http://os-controller:9876

openstack endpoint create --region North\_VN octavia internal http://os-controller:9876

openstack endpoint create --region North\_VN octavia admin http://os-controller:9876

*Bước 2: Cài lại neutron-server container bổ sung thêm neutron-lbaas*

Controller01

Do trước pike thì Octavia cần kết hợp với neutron thông qua plugin neutron-lbaass do đó cần build lại docker neutron-server và run lại trên các controller.

Dockerfiles và các file cần thiết trong thư mục neutron-server-with-lbaas. Lưu ý edit các file cấu hình:

* neutron.conf: Bổ sung thêm cấu hình service\_plugins
* Bổ sung thêm 2 file cấu hình là
  + neutron\_lbaas.conf: Cấu hình service\_auth và service\_providers. Trong phần service\_auth cấu hình user, password Octavia đã tạo ở bước 1
  + services\_lbaas.conf: Cấu hình api Octavia url (Trỏ vào url của haproxy), và chú ý cấu hình thêm tham số request\_poll\_timeout. Mặc định tham số này là 100s, khi đó từ lúc tạo LB đến lúc VM được tạo xong và dịch vụ lên hết mà quá 100s thì trạng thái của LB lỗi => Có thể nâng giá trị này lên tùy môi trường, ví dụ để tạo VM amphora mất 60s, để bật OS, dịch vụ mất 60s => Sẽ lớn hơn 100s => Mặc dù VM amphora tạo ok nhưng quá timeout => trạng thái của LB sẽ lỗi.

Sau đó build và push docker lên registry:

docker build -t ops-services-registry:4000/neutron-server-with-lbaas:ocata neutron/neutron-server-with-lbaas/

docker push ops-services-registry:4000/neutron-server-with-lbaas:ocata

Populate lại databases trên 1 controller (Để tạo thêm các bảng liên quan đến LB trong database neutron)

docker run -d --name neutron-server-with-lbaas --network=host --privileged --user neutron -v /var/log/neutron:/var/log/neutron -v /var/lib/neutron:/var/lib/neutron -v /etc/localtime:/etc/localtime --env-file="./no\_proxy.env" --env-file="./general.env " --env-file="./neutron.env" ops-services-registry:4000/neutron-server-with-lbaas:ocata bootstrap.sh

Sau khi bootrap xong thì xóa container dùng để bootraps

docker ps -a | grep Exit | grep neutron-server-with-lbaas | cut -d ' ' -f 1 | xargs docker rm -f -v

Controller01,02,03

Sau đó stop các docker container neutron-server trên các controller, và chạy container mới

docker stop neutron-server

docker run -d --name neutron-server-with-lbaas --network=host --privileged --user neutron -v /var/log/neutron:/var/log/neutron -v /var/lib/neutron:/var/lib/neutron -v /etc/localtime:/etc/localtime --env-file="./no\_proxy.env" --env-file="./general.env " --env-file="./neutron.env" ops-services-registry:4000/neutron-server-with-lbaas:ocata

Cấu hình thêm các dòng sau của haproxy trên các controller và reload

haproxy tren 3 controller:

listen octavia\_api

bind 192.168.2.137:9876

bind 10.0.0.137:9876

http-request del-header X-Forwarded-Proto

server controller01 10.0.0.129:9876 check inter 2000 rise 2 fall 5

server controller03 10.0.0.131:9876 check inter 2000 rise 2 fall 5

server controller02 10.0.0.130:9876 check inter 2000 rise 2 fall 5

*Bước 3: Build docker Octavia:*

Controller01

Tạo db Octavia và gán quyền truy cập:

CREATE DATABASE octavia;

GRANT ALL PRIVILEGES ON octavia.\* TO 'octavia'@'localhost' IDENTIFIED BY '<password>';

GRANT ALL PRIVILEGES ON octavia.\* TO 'octavia'@'%' IDENTIFIED BY '<password>';

Controller01,02,03

mkdir -p /var/lib/octavia/certs/

mkdir -p /var/log/octavia/

mkdir -p /etc/Octavia

Controller01:

Clone code của Octavia về để tạo cert và build images (Nếu dùng luôn cert và images sẵn có thì có thể bỏ qua bước này)

#clone git ve de tao cert + build imgaes

git clone https://github.com/openstack/octavia/

cd octavia

git checkout -b stable/ocata

Tạo cert hoặc lấy từ compute02 của cloud v1

source bin/create\_certificates.sh /var/lib/octavia/certs/ etc/certificates/openssl.cnf

Tạo images (Hoặc dùng luôn images đã build sẵn của cloud v1)

apt install kpartx qemu

export http\_proxy='http://192.168.5.8:3128'

export https\_proxy='http://192.168.5.8:3128'

cd diskimage-create

pip --proxy http://192.168.5.8:3128 install -r requirements.txt

./diskimage-create.sh

export http\_proxy=''

export https\_proxy=''

Sau khi tạo images xong thì add vào glances và gán tag

openstack image create "amphora-x64-haproxy" --public --container-format bare --file /root/octavia/amphora-x64-haproxy.qcow2

openstack image set --tag amphora amphora-x64-haproxy

Sau khi tạo xong lấy thông tin ID, owner, tag của image glance.

Controller01

Tạo network lb managemt:

* Tạo hoặc dùng sẵn 1 network để phục vụ lb management: Lưu ý network này phải kết nối được đến mạng management của openstack để Octavia\_agent trên VM amphora kết nối với Octavia api, health manager, house keeper.
* Tạo network xong thì tạo subnet.

=> Sau khi tạo lấy ID của network

Tạo security group để phục vụ kết nối management:

neutron security-group-create lb-mgmt-sec-grp

neutron security-group-rule-create --protocol icmp lb-mgmt-sec-grp

neutron security-group-rule-create --protocol tcp --port-range-min 22 --port-range-max 22 lb-mgmt-sec-grp

neutron security-group-rule-create --protocol tcp --port-range-min 9443 --port-range-max 9443 lb-mgmt-sec-grp

Tạo xong thì lấy ID của security group này

Tạo ssh key để quản trị viên có thể ssh vào VM amphora => Tạo xong lấy tên của key.

Controller01,02,03:

Copy file octavia.conf (Trong thư mục octavia) vào thư mục /etc/octavia/ và sửa các tham số về network, cert, images, security group, ssh key, memcache, auth. … Lưu ý tham số controller\_ip\_port\_list nếu cài Octavia trên nhiều controller thì điền dạng list ví dụ 10.0.0.129:5555, 10.0.0.130:5555, 10.0.0.131:5555

Mặc định LB hoạt động ở chế độ single (1 VM amphora/LB). Nếu muốn hoạt động ở chế độ active/standby thì thêm dòng loadbalancer\_topology = ACTIVE\_STANDBY vào [controller\_worker]

Controller01:

Build images Octavia:

docker build -t ops-services-registry:4000/octavia:ocata neutron/octavia/

docker push ops-services-registry:4000/octavia:ocata

Run và chạy populate database:

docker run -d --name octavia-api --network=host --privileged --user root --env-file="./no\_proxy.env" -v /var/log/octavia:/var/log/octavia -v /var/lib/octavia:/var/lib/octavia -v /etc/octavia:/etc/octavia -v /etc/localtime:/etc/localtime ops-services-registry:4000/octavia:ocata octavia-api

Vào bash docker và chay

octavia-db-manage upgrade head

Chạy các docker còn lại:

docker run -d --name octavia-health-manager --network=host --privileged --user root --env-file="./no\_proxy.env" -v /var/log/octavia:/var/log/octavia -v /var/lib/octavia:/var/lib/octavia -v /etc/octavia:/etc/octavia -v /etc/localtime:/etc/localtime ops-services-registry:4000/octavia:ocata octavia-health-manager

docker run -d --name octavia-housekeeping --network=host --privileged --user root --env-file="./no\_proxy.env" -v /var/log/octavia:/var/log/octavia -v /var/lib/octavia:/var/lib/octavia -v /etc/octavia:/etc/octavia -v /etc/localtime:/etc/localtime ops-services-registry:4000/octavia:ocata octavia-housekeeping

docker run -d --name octavia-worker --network=host --privileged --user root --env-file="./no\_proxy.env" -v /var/log/octavia:/var/log/octavia -v /var/lib/octavia:/var/lib/octavia -v /etc/octavia:/etc/octavia -v /etc/localtime:/etc/localtime ops-services-registry:4000/octavia:ocata octavia-worker

Nếu cả 4 docker chạy không lỗi tức là OK.

controller02,03:

Chạy các dockers Octavia:

docker run -d --name octavia-api --network=host --privileged --user root --env-file="./no\_proxy.env" -v /var/log/octavia:/var/log/octavia -v /var/lib/octavia:/var/lib/octavia -v /etc/octavia:/etc/octavia -v /etc/localtime:/etc/localtime ops-services-registry:4000/octavia:ocata octavia-api

docker run -d --name octavia-health-manager --network=host --privileged --user root --env-file="./no\_proxy.env" -v /var/log/octavia:/var/log/octavia -v /var/lib/octavia:/var/lib/octavia -v /etc/octavia:/etc/octavia -v /etc/localtime:/etc/localtime ops-services-registry:4000/octavia:ocata octavia-health-manager

docker run -d --name octavia-housekeeping --network=host --privileged --user root --env-file="./no\_proxy.env" -v /var/log/octavia:/var/log/octavia -v /var/lib/octavia:/var/lib/octavia -v /etc/octavia:/etc/octavia -v /etc/localtime:/etc/localtime ops-services-registry:4000/octavia:ocata octavia-housekeeping

docker run -d --name octavia-worker --network=host --privileged --user root --env-file="./no\_proxy.env" -v /var/log/octavia:/var/log/octavia -v /var/lib/octavia:/var/lib/octavia -v /etc/octavia:/etc/octavia -v /etc/localtime:/etc/localtime ops-services-registry:4000/octavia:ocata octavia-worker

# Hướng dẫn sử dụng:

*Tạo LB:*

neutron lbaas-loadbalancer-create --name lb02 provider

Tạo security group cho LB:

$ neutron security-group-rule-create \

--direction ingress \

--protocol tcp \

--port-range-min 80 \

--port-range-max 80 \

--remote-ip-prefix 0.0.0.0/0 \

lbaas

$ neutron security-group-rule-create \

--direction ingress \

--protocol tcp \

--port-range-min 443 \

--port-range-max 443 \

--remote-ip-prefix 0.0.0.0/0 \

lbaas

$ neutron security-group-rule-create \

--direction ingress \

--protocol icmp \

lbaas

Gán vip port của LB vào security group:

neutron port-update \

--security-group lbaas \

9f8f8a75-a731-4a34-b622-864907e1d556

Tao listener cho lb:

neutron lbaas-listener-create \

--name test-lb02-http \

--loadbalancer lb02 \

--protocol HTTP \

--protocol-port 80

Tạo pool và member cho pool

$ neutron lbaas-pool-create \

--name test-lb02-pool-http \

--lb-algorithm ROUND\_ROBIN \

--listener test-lb02-http \

--protocol HTTP

$ neutron lbaas-member-create \

--name test-lb02-http-member-1 \

--subnet provider \

--address 192.168.2.180 \

--protocol-port 80 \

test-lb02-pool-http

$ neutron lbaas-member-create \

--name test-lb02-http-member-2 \

--subnet provider \

--address 192.168.2.176 \

--protocol-port 80 \

test-lb02-pool-http

Tạo healthcheck cho pool (optional)

neutron lbaas-healthmonitor-create \

--name test-lb02-http-monitor \

--delay 5 \

--max-retries 2 \

--timeout 10 \

--type HTTP \

--pool test-lb02-pool-http

Nếu quá trình tạo LB ok thì sẽ có 2 máy ảo vm amphora, đồng thời xem trạng thái của LB bằng lệnh neutron lbaas-loadbalancer-list sẽ có trạng thái active và địa chỉ VIP.

Từ ngoài có thể truy cập vào VIP để truy cập dịch vụ.

Có thể update thêm các tham số về thuật toán LB, session sticky …. Thông qua các lệnh neutron lbaas-\*

Trong trường hợp nếu 2 máy ảo VM amphora tạo ok nhưng trạng thái LB là error thì có thể do timeout trong services\_lbaas.conf đặt thấp, cần tăng giá trị này lên và thử lại.

*Cách kiểm tra cấu hình trong máy ảo amphora*

SSH vào máy ảo Amphora bằng được lb management và ssh key đã tạo.

Để kiểm tra cấu hình real IP và virtual IP thì switch sang namespace amphora-haproxy

ip netns exec amphora-haproxy ifconfig

Kiểm tra cấu hình haproxy hiện tại:

ps -ef | grep haproxy

=> Sẽ thấy tiến trình hayproxy và đường dẫn file cấu hình, ví dụ:

/var/lib/octavia/d21895f1-f04a-4332-9490-578f01da86c7/haproxy.cfg

Kiểm tra cấu hình keepalived (chỉ có khi chạy ở topo active/standby):

/var/lib/octavia/vrrp/octavia-keepalived.conf

*Kiểm tra hoạt động của health manager và house keeper:*

Thử disable card mạng management trên 1 VM amphora đi:

- nova list => Lấy id cua LB

- neutron port-list -- --device-id <device-id> => Lấy ID của port management

- neutron port-update <port-id> --admin-state-up False => Disable port management

Khi đấy trên logs của docker octavia-health-manager sẽ phát hiện LB die => Ra lệnh cho Nova xóa VM cũ đi, ra lệnh cho neutron detach IP real và ip virtual khỏi VM cũ, ra lệnh cho nova tạo lại 1 VM mới và ra lệnh cho neutron vẫn dùng IP virtual, ra lệnh cho neutron attach lại ip real và virtual ip cũ.

Trên log của docker octavia-housekeeping sẽ có log xóa bản ghi DB của VM cũ.